Configuring the Device and Chassis Manager

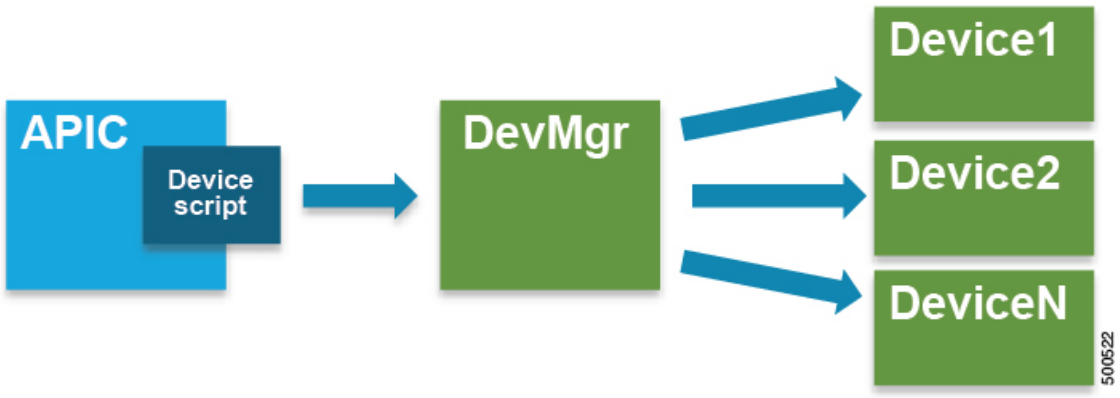
- About Device Managers and Chassis Managers, on page 1
- Device Manager and Chassis Manager Behavior, on page 4
- Creating Devices Using the GUI, on page 5
- Creating a Chassis Using the GUI, on page 5
- Example XML for Device Managers and Chassis Managers, on page 6
- Device and Chassis Callouts, on page 8

About Device Managers and Chassis Managers

A device manager serves as a single point of configuration for a set of clusters in a Cisco Application Centric Infrastructure (ACI) fabric. The administration or operational state is presented in the native GUI of the devices. A device manager handles configuration on individual devices and enables you to simplify the configuration in the Application Policy Infrastructure Controller (APIC). You create a template in device manager, then populate the device manager with instance-specific values from the APIC, which requires only a few values.

The following figure illustrates a device manager controlling multiple devices in a cluster:

Figure 1: Controlling Devices with a Device Manager

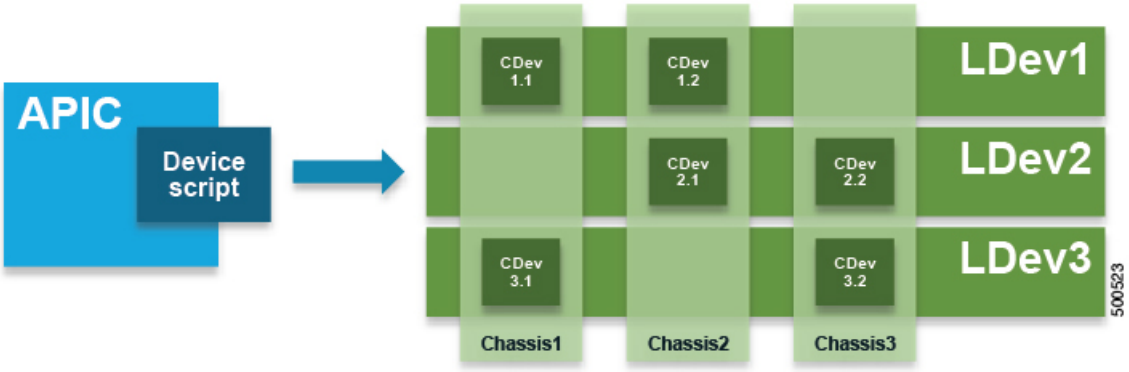


A chassis manager is a physical or virtual "container" of processing resources. A chassis manager supports a number of virtual service devices that are represented as `CDev` objects. A chassis manager handles networking, while `CDev` handles processing. A chassis manager enables the on-demand creation of virtual processing nodes.

For a virtual device, some parts of a service (specifically the VLANs) must be applied to the chassis rather than to the virtual machine. To accomplish this, the chassis management IP address and credentials must be included in callouts.

The following figure illustrates a chassis manager acting as a container of processing resources:

Figure 2: Controlling Devices with a Device Manager

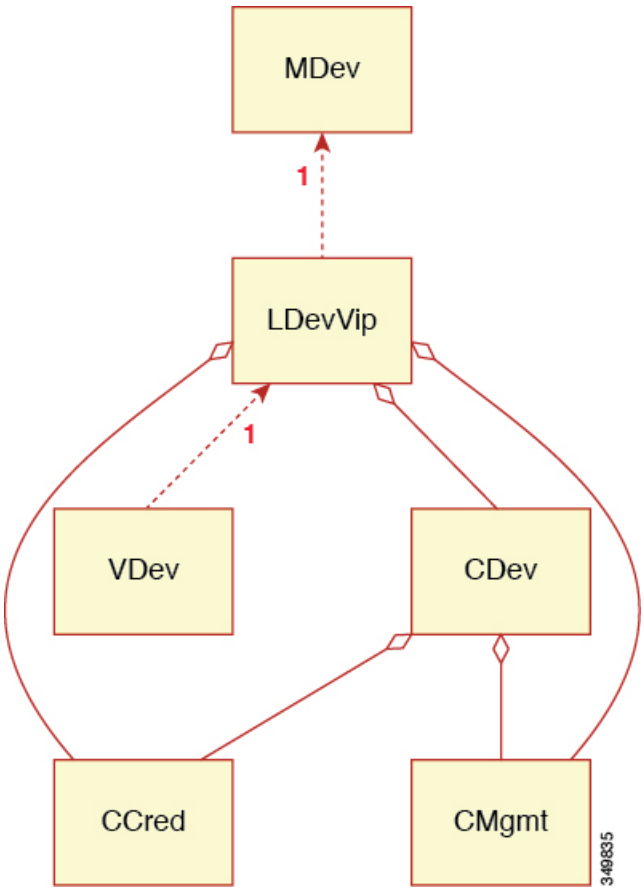


Without a device manager or chassis manager, the model for service devices contains the following key managed objects:

- `MDev`—Represents a device type (vendor, model, version).
- `LDevVIP`—Represents a cluster, a set of identically configured devices for Cold Standby. Contains `CMgmt` and `CCred` for access to the device.
- `CDev`—Represents a member of a cluster, either physical or virtual. Contains `CMgmt` and `CCred` for access to the device.
- `VDev`—Represents a context on a cluster, similar to a virtual machine on a server.

The following figure illustrates the model for the key managed objects, with `CMgmt` (management connectivity) and `CCred` (credentials) included:

Figure 3: Managed Object Model Without a Device Manager or Chassis Manager



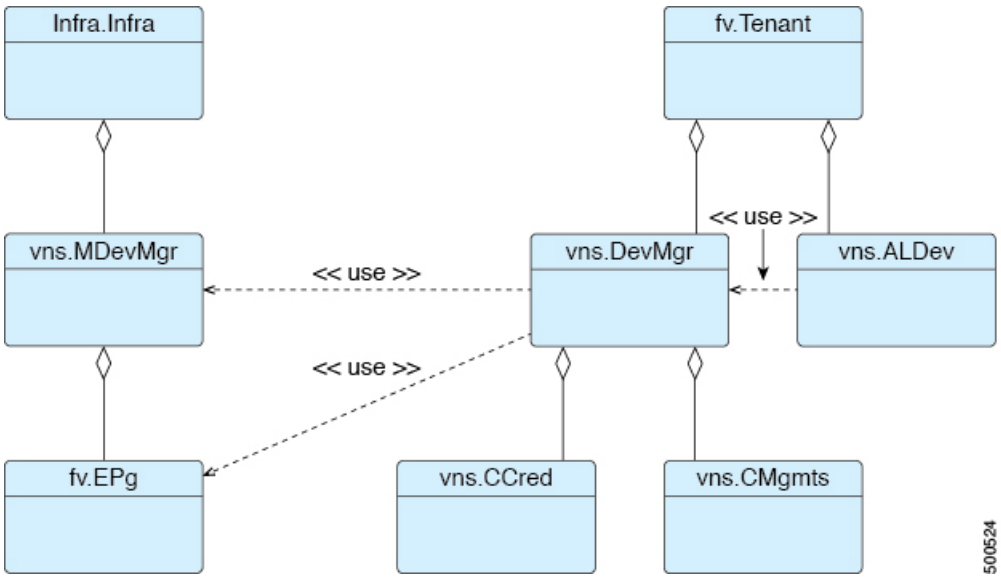
`CMgmt` (host + port) and `CCred` (username + password) allow the script to access the device and cluster.

A device manager and chassis manager adds the ability to control the configuration of clusters and devices from a central management station. The chassis adds a parallel hierarchy to the `MDev` object and `ALDev` object to allow a `CDev` object to be tagged as belonging to a specific chassis. The following managed objects are added to the model to support the device and chassis manager concept:

- `MDevMgr`—Represents a type of device manager. An `MDevMgr` can manage a set of different `MDev`s, which are typically different products from the same vendor.
- `DevMgr`—Represents a device manager. Access to the manager is provided using the contained `CMgmt` and `CCred` managed objects. Each cluster can be associated with only one `DevMgr`.
- `MChassis`—Represents a type of chassis. This managed object is typically included in the package.
- `Chassis`—Represents a chassis instance. It contains the `CMgmt` and `CCred`[Secret] managed objects to provide connectivity to the chassis.

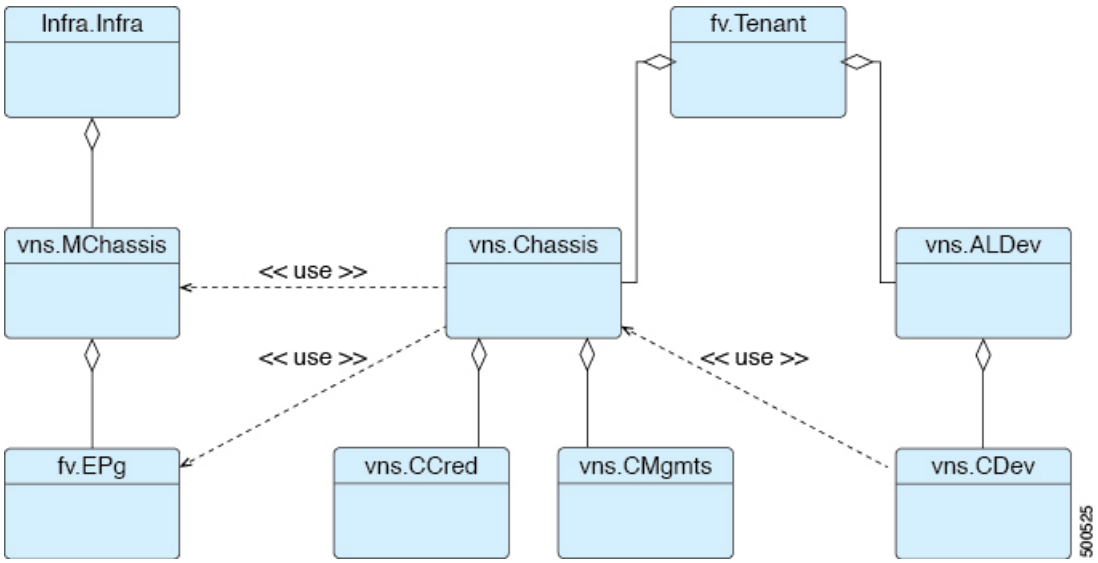
The following figure illustrates the device manager object model:

Figure 4: Device Manager Object Model



The following figure illustrates the chassis manager object model:

Figure 5: Chassis Manager Object Model



Device Manager and Chassis Manager Behavior

The following behavior applies regarding device managers and chassis managers:

- The `DevMgr` object is not required. If there is no relation from `LDevVip` to `DevMgr`, then the system performs callouts without the device manager being defined.

- `Policymgr` performs a sanity check to ensure that the relation from `LDevVip` to `MDev` matches one relation path from `LDevVip` through `DevMgr` and `MDevMgr` to `MDev`. A fault is raised if this is not the case, which prevents further callouts.
- If a relation from `LDevVip` to `DevMgr`, from `DevMgr` to `MDevMgr`, or from `MDevMgr` to the correct `MDev` is added or changed, then the cluster is reset and configured from scratch
- The `Chassis` object is not required. If there is no relation from `CDev` to `Chassis`, then the system performs callouts without the chassis being defined.
- `Policymgr` performs a sanity check to ensure that the relation from `CDev` through `LDevVip` to `MDev` matches one relation path from `CDev` through `Chassis` and `MChassis` to `MDev`. A fault is raised if this is not the case, which prevents further callouts.
- If a relation from `CDev` to `Chassis`, from `Chassis` to `MChassis`, or from `MChassis` to the correct `MDev` is added or changed, then the cluster is reset and configured from scratch

Creating Devices Using the GUI

You can create a device manager for a tenant using the GUI.

Step 1 On the menu bar, choose **Tenants** > **All Tenants**.
Step 2 In the Work pane, double click a tenant's name.
Step 3 In the Navigation pane, choose **Tenant** ***tenant_name*** > **Services** > **L4-L7** > **Device Managers**.
Step 4 In the Work pane, choose **Actions** > **Create Device Manager**.
Step 5 In the **Create Device Manager** dialog box, fill in the fields as required.

Note Policy Based Redirect configuration for Layer 1 or Layer 2 is selected in **Device Manager Type**.

Step 6 Click **Submit**.

Creating a Chassis Using the GUI

You can create a chassis for a tenant using the GUI.

Step 1 On the menu bar, choose **Tenants** > **All Tenants**.
Step 2 In the Work pane, double click a tenant's name.
Step 3 In the Navigation pane, choose **Tenant** ***tenant_name*** > **Services** > **L4-L7** > **Chassis**.
Step 4 In the Work pane, choose **Actions** > **Create Chassis**.
Step 5 In the **Create Chassis** dialog box, fill in the fields as required.
Step 6 Click **Submit**.

Example XML for Device Managers and Chassis Managers

The example XML in the following sections assumes that the `Insieme` package has been loaded, which provides the `uni/infra/mDev-Insieme-Generic-1.0` distinguished name.

Example XML for Creating the MDevMgr Object

The `MDevMgr` object is analogous with the `MDev` object and has vendor, model, and version as the naming properties. Multiple `MDevMgrToMDev` relations can be created if the manager can manage different types of clusters. The following example XML creates the `MDevMgr` object:

```
<polUni>
  <infraInfra>
    <vnsMDevMgr
        vendor="Insieme"
        model="DevMgr"
        version="1.0"
        >
      <vnsRsMDevMgrToMDev tDn="uni/infra/mDev-Insieme-Generic-1.0"/>
    </vnsMDevMgr>
  </infraInfra>
</polUni>
```

The following example XML creates the `MDevMgr` object within the `tenant1` tenant:

```
<polUni>
    <fvTenant name="tenant1">
      <vnsDevMgr name="Foo">
        <vnsCMgmts
            host="10.10.11.11"
            port="1234"/>
        <vnsCMgmts
            host="10.10.11.12"
            port="1234"/>
        <vnsCMgmts
            host="10.10.11.13"
            port="1234"/>
        <vnsCCred name="username" value="admin"/>
        <vnsCCredSecret name="password" value="letmein"/>
        <vnsRsDevMgrToMDevMgr tDn="uni/infra/mDevMgr-Insieme-DevMgr-1.0"/>
    </vnsDevMgr>
  </fvTenant>
</polUni>
```

Example XML for Associating an LDevVip Object With a DevMgr Object

Association of the `LDevVip` object with the `DevMgr` object is done by creating a relation from `LDevVip` to `DevMgr`, as shown in the following example XML:

```
<polUni>
    <fvTenant name="tenant1">
        <vnsLDevVip name="InsiemeCluster" devtype="VIRTUAL">
            ...
            <vnsRsMDevAtt tDn="uni/infra/mDev-Insieme-Generic-1.0"/>
            <vnsRsALDevToDevMgr tDn="uni/tn-tenant1/devMgr-Foo"/>
            ...
        </vnsLDevVip>
```

```
    </fvTenant>
</polUni>
```

Example XML for Creating the MChassis Object

The `MChassis` object is analogous with the `MDev` object and has vendor, model, and version as the naming properties. A `MChassisToMDev` relation determines the device type. The following example XML creates the `MChassis` object:

```
<polUni>
  <infraInfra>
    <vnsMChassis vendor="Insieme" model="DevMgr" version="1.0">
      <vnsRsMChassisToMDev tDn="uni/infra/mDev-Insieme-Generic-1.0"/>
    </vnsMChassis>
  </infraInfra>
</polUni>
```

Example XML for Creating the Chassis Object

The following example XML creates the `Chassis` object:

```
<polUni>
  <fvTenant name="tenant1">
    <vnsChassis name="Foo">
        <vnsCMgmts
            host="10.10.11.11"
            port="1234"/>
        <vnsCMgmts
            host="10.10.11.12"
            port="1234"/>
        <vnsCMgmts
            host="10.10.11.13"
            port="1234"/>
      <vnsCCred name="username" value="admin"/>
      <vnsCCredSecret name="password" value="letmein"/>
      <vnsRsChassisToMChassis tDn="uni/infra/mChassis-Insieme-DevMgr-1.0"/>
    </vnsChassis>
  </fvTenant>
</polUni>
```

Example XML for Associating an CDev Object With a Chassis Object

Association of the `CDev` object with the `Chassis` object is done by creating a relation from `CDev` to `Chassis`, as shown in the following example XML:

```
<polUni>
    <fvTenant name="tenant1">
        <vnsLDevVip name="InsiemeCluster" devtype="VIRTUAL">
           ...
           <vnsCDev name="Generic1" devCtxLbl="C1">
              ...
              <vnsRsCDevToChassis tnVnsChassisName="Foo"/>
            </vnsCDev>
            <vnsRsALDevToDevMgr tDn="uni/tn-coke/devMgr-Foo"/>
           ...
        </vnsLDevVip>
    </fvTenant>
</polUni>
```

Device and Chassis Callouts

The following sections contain device, cluster, and service callout examples that include parameters for the device and chassis manager. The parameters are added to all callouts.

Example deviceValidate Callout for a Device

In the following example `deviceValidate` callout, the device-specific code is shown in bold:

```
2014-10-03 17:38:51,035 DEBUG 140230105585408 [42.42.42.101, 0]:  deviceValidate
{'args': ('1.0',),
 'device': {'creds': {'password': '<hidden>', 'username': 'nsroot'},
            'host': '42.42.42.101',
            'manager': {'creds': {'username': 'admin', 'password': '<hidden>'},
                        'hosts': {'10.10.11.11': {'port': 1234},
                                  '10.10.11.12': {'port': 1234},
                                  '10.10.11.13': {'port': 1234}},
                        'name': 'Foo'},
            'port': 80,
            'version': '1.0',
            'virtual': True}}
```

Example deviceAudit Callout for a Device

In the following example `deviceAudit` callout, the device-specific code is shown in bold:

```
2014-10-03 17:38:56,072 DEBUG 140230088800000 [42.42.42.100, 2]:  deviceAudit
{'args': ({(11, '', 'ext'): {'label': 'in', 'state': 0},
    (11, '', 'int'): {'label': 'out', 'state': 0}},
    {(4, 'oneFolder', 'foo'): {'ackedState': 0,
        'state': 0,
        'transaction': 0,
        'value': {(5, 'oneParam', 'foo'): {'ackedState': 0,
            'state': 0,
            'transaction': 0,
            'value': 'bar'}}}}),
'device': {'creds': {'password': '<hidden>', 'username': 'nsroot'},
    'host': '42.42.42.100',
    'manager': {'creds': {'username': 'admin', 'password': '<hidden>'},
                        'hosts': {'10.10.11.11': {'port': 1234},
                                  '10.10.11.12': {'port': 1234},
                                  '10.10.11.13': {'port': 1234}},
                        'name': 'Foo'},
    'port': 80,
    'version': '1.0',
    'virtual': True}}
```

Example clusterAudit Callout for a Device

In the following example `clusterAudit` callout, the device-specific code is shown in bold:

```
2014-10-03 17:39:01,097 DEBUG 140229734295296 [42.42.42.99, 4]:  clusterAudit
{'args': ({(12, '', 'ext'): {'cifs': {'Generic1': 'ext', 'Generic2': 'ext'},
                             'label': 'in',
                             'state': 0},
           (12, '', 'inside'): {'cifs': {'Generic1': 'ext',
                                         'Generic2': 'ext'},
```

```
                                'label': 'in',
                                'state': 0},
              (12, '', 'int'): {'cifs': {'Generic1': 'int', 'Generic2': 'int'},
                                'label': 'out',
                                'state': 0},
              (12, '', 'outside'): {'cifs': {'Generic1': 'int',
                                             'Generic2': 'int'},
                                    'label': 'out',
                                    'state': 0}},
             {}),
 'device': {'creds': {'password': '<hidden>', 'username': 'nsroot'},
            'devs': {'Generic1': {'creds': {'password': '<hidden>',
                                            'username': 'nsroot'},
                                  'host': '42.42.42.100',
                                  'port': 80,
                                  'virtual': True},
                     'Generic2': {'creds': {'password': '<hidden>',
                                            'username': 'nsroot'},
                                  'host': '42.42.42.101',
                                  'port': 80,
                                  'virtual': True}},
            'host': '42.42.42.99',
            'manager': {'creds': {'username': 'admin', 'password': '<hidden>'},
                        'hosts': {'10.10.10.11': {'port': 1234},
                                  '10.10.10.12': {'port': 1234},
                                  '10.10.10.13': {'port': 1234}},
                        'name': 'Foo'},
            'port': 80,
            'version': '1.0',
            'virtual': True}}
```

Example serviceAudit Callout for a Device

In the following example `serviceAudit` callout, the device-specific code is shown in bold:

```
2014-10-03 17:39:06,169 DEBUG 140229725902592 [42.42.42.99, 5]:  serviceAudit
{'args': ({(0, '', 4474): {'ackedState': 0,
                           'state': 2,
                           'transaction': 0,
                           'txid': 10000,
                           'value': {(1, '', 5787): {
                               ...
                           }},
 'device': {'creds': {'password': '<hidden>', 'username': 'nsroot'},
            'devs': {'Generic1': {'creds': {'password': '<hidden>',
                                            'username': 'nsroot'},
                                  'host': '42.42.42.100',
                                  'port': 80,
                                  'virtual': True},
                     'Generic2': {'creds': {'password': '<hidden>',
                                            'username': 'nsroot'},
                                  'host': '42.42.42.101',
                                  'port': 80,
                                  'virtual': True}},
            'host': '42.42.42.99',
            'manager': {'creds': {'username': 'admin', 'password': '<hidden>'},
                        'hosts': {'10.10.11.11': {'port': 1234},
                                  '10.10.11.12': {'port': 1234},
                                  '10.10.11.13': {'port': 1234}},
                        'name': 'Foo'},
            'port': 80,
            'version': '1.0',
            'virtual': True}}
```

Example deviceValidate Callout for a Chassis

In the following example `deviceValidate` callout, the chassis-specific code is shown in bold:

```
2014-11-13 19:33:16,066 DEBUG 140719921972992 [42.42.42.101, 0]: request: deviceValidate
 {'args': ('1.0',),
  'device': {'chassis': {'creds': {'username': 'admin', 'password': '<hidden>'},
                         'hosts': {'10.10.11.11': {'port': 1234},
                                   '10.10.11.12': {'port': 1234},
                                   '10.10.11.13': {'port': 1234}},
                         'name': 'Foo'},
             'creds': {'username': 'nsroot', 'pasword': '<hidden>'},
             'host': '42.42.42.100',
             'port': 80,
             'virtual': True}}
```

Example deviceAudit Callout for a Chassis

In the following example `deviceAudit` callout, the chassis-specific code is shown in bold:

```
2014-10-03 17:38:56,072 DEBUG 140230088800000 [42.42.42.100, 2]:  deviceAudit
 {'args': ({(11, '', 'ext'): {'label': 'in', 'state': 0},
            (11, '', 'int'): {'label': 'out', 'state': 0}},
           {(4, 'oneFolder', 'one'): {'ackedState': 0,
                         'state': 0,
                         'transaction': 0,
                         'value': {(5, 'oneParam', 'one'): {'ackedState': 0,
                                   'state': 0,
                                   'transaction': 0,
                                   'value': 'foo'}}}}),
  'device': {'chassis': {'creds': {'username': 'admin', 'password': '<hidden>'},
                         'hosts': {'10.10.11.11': {'port': 1234},
                                   '10.10.11.12': {'port': 1234},
                                   '10.10.11.13': {'port': 1234}},
                         'name': 'Foo'},
             'creds': {'password': '<hidden>', 'username': 'nsroot'},
             'host': '42.42.42.101',
             'manager': {'creds': {'username': 'admin', 'password': '<hidden>'},
                         'hosts': {'10.10.10.11': {'port': 1234},
                                   '10.10.10.12': {'port': 1234},
                                   '10.10.10.13': {'port': 1234}},
                         'name': 'Foo'},
             'port': 80,
             'version': '1.0',
             'virtual': True}}
```

Example clusterAudit Callout for a Chassis

In the following example `clusterAudit` callout, the chassis-specific code is shown in bold:

```
2014-10-03 17:39:01,097 DEBUG 140229734295296 [42.42.42.99, 4]:  clusterAudit
 {'args': ({(12, '', 'ext'): {'cifs': {'Generic1': 'ext', 'Generic2': 'ext'},
                              'label': 'in',
                              'state': 0},
            (12, '', 'inside'): {'cifs': {'Generic1': 'ext',
                                          'Generic2': 'ext'},
                                 'label': 'in',
                                 'state': 0},
            (12, '', 'int'): {'cifs': {'Generic1': 'int', 'Generic2': 'int'},
                              'label': 'out',
                              'state': 0},
```

```
          (12, '', 'outside'): {'cifs': {'Generic1': 'int',
                                         'Generic2': 'int'},
                                'label': 'out',
                                'state': 0}},
         {}),
 'device': {'creds': {'password': '<hidden>', 'username': 'nsroot'},
            'devs': {'Generic1': {'chassis': {'creds': {'password': '<hidden>',
                                                        'username': 'admin'},
                                              'hosts': {'10.10.11.11': {'port': 1234},
                                                        '10.10.11.12': {'port': 1234},
                                                        '10.10.11.13': {'port': 1234}},
                                              'name': 'Foo'},
                                  'creds': {'username': 'nsroot', 'pasword': '<hidden>'},
                                  'host': '42.42.42.100',
                                  'port': 80,
                                  'virtual': True},
                     'Generic2': {'chassis': {'creds': {'password': '<hidden>',
                                                        'username': 'admin'},
                                              'hosts': {'10.10.11.11': {'port': 1234},
                                                        '10.10.11.12': {'port': 1234},
                                                        '10.10.11.13': {'port': 1234}},
                                              'name': 'Foo'},
                                  'creds': {'username': 'nsroot', 'pasword': '<hidden>'},
                                  'host': '42.42.42.101',
                                  'port': 80,
                                  'virtual': True}},
            'host': '42.42.42.99',
            'manager': {'creds': {'password': '<hidden>', 'username': 'admin'},
                        'hosts': {'10.10.11.11': {'port': 1234},
                                  '10.10.11.12': {'port': 1234},
                                  '10.10.11.13': {'port': 1234}},
                        'name': 'Foo'},
            'port': 80,
            'version': '1.0',
            'virtual': True}}
```

Example serviceAudit Callout for a Chassis

In the following example `serviceAudit` callout, the chassis-specific code is shown in bold:

```
2014-10-03 17:39:06,169 DEBUG 140229725902592 [42.42.42.99, 5]:  serviceAudit
 {'args': ...,
  'device': {'creds': {'password': '<hidden>', 'username': 'nsroot'},
             'devs': {'Generic1': {'chassis': {'creds': {'username': 'admin',
                                                         'password': '<hidden>'},
                                               'hosts': {'10.10.11.11': {'port': 1234},
                                                         '10.10.11.12': {'port': 1234},
                                                         '10.10.11.13': {'port': 1234}},
                                               'name': 'Foo'},
                                   'creds': {'username': 'nsroot',
                                             'pasword': '<hidden>'},
                                   'host': '42.42.42.100',
                                   'port': 80,
                                   'virtual': True},
                      'Generic2': {'chassis': {'creds': {'username': 'admin',
                                                         'password': '<hidden>'},
                                               'hosts': {'10.10.11.11': {'port': 1234},
                                                         '10.10.11.12': {'port': 1234},
                                                         '10.10.11.13': {'port': 1234}},
                                               'name': 'Foo'},
                                   'creds': {'username': 'nsroot',
                                             'pasword': '<hidden>'},
```

```
                      'host': '42.42.42.101',
                      'port': 80,
                      'virtual': True}},
'host': '42.42.42.99',
'manager': {'creds': {'username': 'admin', 'password': '<hidden>'},
            'hosts': {'10.10.11.11': {'port': 1234},
                      '10.10.11.12': {'port': 1234},
                      '10.10.11.13': {'port': 1234}},
            'name': 'Foo'},
'port': 80,
'version': '1.0',
'virtual': True}}
```